J Algebr Comb (2014) 40:743–770
DOI 10.1007/s10801-014-0505-x

A symmetry property for q-weighted Robinson–Schensted and other branching insertion algorithms

Yuchen Pei

Received: 6 November 2013 / Accepted: 29 January 2014 / Published online: 25 March 2014

Abstract In [19], a q-weighted version of the Robinson–Schensted algorithm was introduced. In this paper, we show that this algorithm has a symmetry property analogous to the well-known symmetry property of the usual Robinson–Schensted algorithm. The proof uses a generalisation of the growth diagram approach introduced by Fomin [5–8]. This approach, which uses 'growth graphs', can also be applied to a wider class of insertion algorithms which have a branching structure, including some of the other q-weighted versions of the Robinson–Schensted algorithm which have recently been introduced by Borodin–Petrov [2].

Keywords Robinson–Schensted algorithm · Growth diagram · Q-analogue · Permutation

Mathematics Subject Classification 05C85 · 05E10 · 05E05

1 Introduction

In [19], a q-weighted version of the Robinson–Schensted (RS) algorithm was introduced. In this paper, we show that this algorithm enjoys a symmetry property analogous to the well-known symmetry property of the RS algorithm. The proof uses a generalisation of the growth diagram approach introduced by [5–8].

The insertion algorithm we consider in this paper is based on column insertion, but the technique applies to any insertion algorithm belonging to a certain class of 'branching insertion algorithms', as described in Sect. 7 below. For example, [2]

Y. Pei (✉)
Mathematics Institute, University of Warwick, Coventry, UK
e-mail: y.pei@warwick.ac.uk

have recently introduced a q-weighted version of the row insertion algorithm, which is defined similarly to the column insertion version of [19]; they also consider a wider family of such algorithms (and, more generally, dynamics on Gelfand–Tsetlin patterns), some of which fall into the framework considered in the present paper, and can similarly be shown to have the symmetry property. We discuss such extensions in Sect. 7 below.

The Robinson–Schensted algorithm is a combinatorial algorithm which was introduced by Robinson [1] and Schensted [22]. It has wide applications in representation theory and probability theory, e.g. last passage percolation, totally asymmetric simple exclusion process/corner growth model, random matrix theory [12], queues in tandem [17] and more.

There are two versions of RS algorithms, the row insertion and column insertion version. In most of this paper, we deal with column insertion and its q-version. The RS algorithm transforms a word to a tableau pair of the same shape. A word can be treated as a path, and hence a random word corresponds to a random walk. When taking such a random walk, the shape of the output tableaux is a Markov chain, whose transition kernel is related to the Schur symmetric functions [16]. A geometric generalisation transforms a Brownian motion with drift to a Markov process whose generator is related to $\mathsf{GL}(n,\mathbb{R})$-Whittaker functions [18], which are eigen functions of the quantum Toda chain [14]. The q-Whittaker functions on the one hand are a generalisation of the Schur symmetric functions and a specialisation of Macdonald (q,t)-symmetric functions when $t=0$ [15], and on the other hand are eigen functions of the q-deformed quantum Toda chain [4,20]. When $q\to 0$ they become the Schur functions, and when $q\to 1$ with a proper scaling [10] they converge to Whittaker functions. In the spirit of this connection, a q-weighted RS algorithm was formulated in [19], which transforms the random walk to a Markov chain that is related to q-Whittaker functions.

As is expected, the algorithm degenerates to the usual RS algorithm when $q\to 0$. Part of the random P-tableau also has q-TASEP dynamics [19], the latter introduced in [24], just as the same part of the random P-tableau of usual RS algorithm has TASEP dynamics [17].

Recently, a q-version of the RS row insertion algorithm was also introduced in [2]. It denegerates to the usual RS algorithm with row insertion when $q\to 0$.

In this paper, we show that both algorithms enjoy a symmetry property analogous to the well-known symmetry property of the RS algorithm. Basically, the symmetry property for the RS algorithms restricted to permutation inputs is the property that the output tableau pair is interchanged if the permutation is inversed. Knuth [13] generalised the usual row insertion algorithm to one which takes matrix input, which we refer to as Robinson–Schensted–Knuth (RSK) algorithm. For this algorithm, the symmetry property is that the output tableau pair is interchanged if the matrix is transposed. Note that the matrix becomes the permutation matrix when the RSK algorithm is restricted to permutation, and hence the transposition of the matrix corresponds to inversion of the permutation. Burge gives a similar generalisation of the column insertion algorithm [3], which we refer to as Burge's algorithm.

The symmetry property for the usual RS algorithm is normally discussed in the literature for the row insertion. However, in the permutation case, the output tableau pair for row insertion is simply the transposition of the pair for column insertion. Therefore, proofs of the symmetry property can be translated to column insertion instantly. In the matrix input case, the Burge and RSK algorithms are also closely related so that proofs can be extended from one to the other naturally.

For a proof of the symmetry property, one can see e.g. [9,21,23]. There are a few different approaches to deal with it. One is the Viennot diagram [25] which provides a nice geometric construction of the RS algorithms. In the matrix input case, there are two approaches which reduce to the Viennot diagram approach when restricted to permutations. One is the antichain or the inversion digraph construction of the RSK algorithm (which should extend naturally to the Burge algorithm) due to [13]; the other (for both RSK and Burge) is Fomin's matrix-ball construction of [9].

Another method is the growth diagram technique due to Fomin [5–8]. It can be generalised to the RSK and Burge algorithms. Greene's theorem [11] gives a proof by showing the relation between the lengths of the longest subsequences of the input and the shape of the output. However, the growth diagram approach can be thought of as a fast construction to calculate these lengths.

Among all these techniques, the special structure of growth diagram can be extended to a class of algorithms what we will call branching algorithms, which include the q-weighted column and row insertion algorithms. Therefore, it is this approach which we use in this paper. For a simple description of the technique for usual row insertion see e.g. [23], whose column version will be shown in Sect. 3.

The rest of the paper is organised in the following way. In Sect. 2, we recall the insertion rule of a letter into a tableau for the usual RS algorithm (with column insertion). We describe it in a way that suits the growth diagram. In Sect. 3, we describe the insertion rule for a word and state the symmetry property for the RS algorithm applied to permutations. In Sect. 4 and 5, we describe the q-weighted insertion algorithm for letters and words in a way which is different, but equivalent to the definition given in [19]. In Sect. 6, we state and prove the symmetry property in the q-case. Finally, in Sect. 7, we prove the symmetry property for the row insertion algorithm and more generally branching algorithms.

2 Classical Robinson–Schensted algorithm

A partition $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k) \in W = \{(a_1, a_2, \ldots) \in \bigcup_{k=1}^{\infty} \mathbb{N}_{\geq 0}^k, a_1 \geq a_2 \geq \ldots\}$ is a vector of weakly decreasing non-negative integer entries. Denote by $l(\lambda)$ the number of positive entries of λ and $|\lambda| = \lambda_1 + \cdots + \lambda_{l(\lambda)}$ the size of the partition. We say λ is a partition of n if $|\lambda| = n$, which we denote by $\lambda \vdash n$. A Young tableau P with shape λ is a left-aligned array of $l(\lambda)$ rows of positive integers such that the entries are strictly increasing along each column and weakly increasing along each

row, and such that the length of the jth row is λ_j. For example, below is a tableau with shape $(4, 3, 2, 2)$.

$$\begin{array}{llll} 1 & 1 & 3 & 4 \\ 3 & 5 & 8 & \\ 6 & 7 & & \\ 8 & 8 & & \end{array} \tag{1}$$

We denote by $\mathrm{sh}P$ the shape of tableau P and $\mathcal{T}_\ell$ the set of all tableaux with entries no greater than ℓ. Denote $[n] = \{1, 2, \ldots, n\}$ for a positive integer n. A standard tableau Q is a tableau with distinct entries from $[|\mathrm{sh}Q|]$. For example, below is a standard tableau with shape $(4, 3, 2, 2)$.

$$\begin{array}{llll} 1 & 3 & 4 & 7 \\ 2 & 5 & 8 & \\ 6 & 10 & & \\ 9 & 11 & & \end{array}$$

We denote by $\mathcal{S}_n$ the set of standard tableaux with shape of size n. For example, the above tableau is an element of $\mathcal{S}_{11}$.

For a tableau P, we denote by P^k its subtableau containing all entries no greater than k and call it the kth subtableau of P. For example, the 6th subtableau of the tableau shown in (1) is

$$\begin{array}{llll} 1 & 1 & 3 & 4 \\ 3 & 5 & & \\ 6 & & & \end{array}$$

A tableau P can be identified by the shape of its subtableaux, which we usually denote by $\lambda^k = \mathrm{sh}P^k$. Also, let $\lambda^0 = \emptyset$ be the empty partition. Evidently, a tableau has only finitely many different λ^i's and there exists an ℓ such that $\lambda^i = \mathrm{sh}P$ for $i \geq \ell$. We call λ^i the ith shape of P. We can identify P with these shapes and write $P = \lambda^0 \prec \lambda^1 \prec \lambda^2 \prec \cdots \prec \lambda^\ell$, where '$\prec$' is an interlacing relation: for $a = (a_1, a_2, \ldots)$ and $b = (b_1, b_2, \ldots)$, $a \prec b$ means $b_1 \geq a_1 \geq b_2 \geq a_2 \geq \ldots$. For example, the tableau P in (1) is identified as

$$P = \emptyset \prec 2 \prec 2 \prec 31 \prec 41 \prec 42 \prec 421 \prec 422 \prec 4322.$$

The basic operation of the RS algorithm is to insert a letter $k \in \mathbb{N}_+ := \mathbb{N}\backslash\{0\}$ into a tableau $P = \emptyset \prec \lambda^1 \prec \lambda^2 \prec \cdots \prec \lambda^\ell$ and produce a new tableau $\tilde{P} = \emptyset \prec \tilde{\lambda}^1 \prec \tilde{\lambda}^2 \prec \cdots \prec \tilde{\lambda}^\ell$. To do this, we first find the lowest row in λ^k such that appending a box at the end of that row would preserve the interlacement between the $k-1$th shape and the kth shape, and append the box to that row. Suppose the row has index j_k, then we find the lowest row in λ^{k+1} that is no lower than row j_k such that appending a box at the end of that row would preserve the interlacement between the kth shape and the $k+1$th shape, append the box to that row and so on and so forth. More precisely, define

$$j_{k-1} = k; \quad j_i = \max(\{j \le j_{i-1} : \lambda^{i-1}_{j-1} > \lambda^i_j\} \cup \{1\}), \qquad i \ge k.$$

Then, the new tableau $\tilde{P}$ is defined by

$$\tilde{\lambda}^i = \begin{cases} \lambda^i, & \text{if } i < k; \\ \lambda^i + \mathbf{e}_{j_i}, & \text{otherwise.} \end{cases}$$

where $\mathbf{e}_j$ is the jth standard basis of $\mathbb{R}^{\mathbb{N}_+}$.

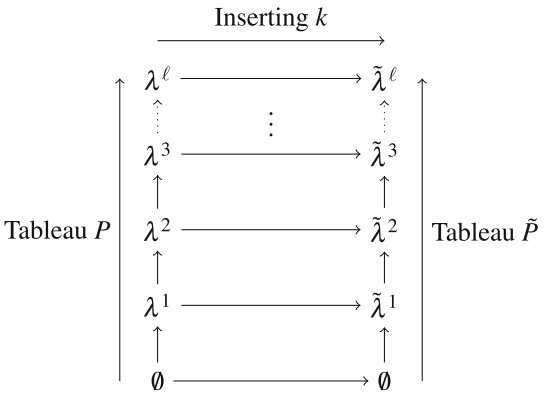
For example, if we insert a 6 into the tableau shown in (1), the insertion process is shown as follows:

$$\begin{array}{l} \hat{5}\,\hat{5}\,\hat{5}\,\hat{5} \\ \hat{5}\,\hat{5} \\ \hat{6}\,\hat{\mathbf{6}} \end{array} \quad \to \quad \begin{array}{l} \hat{6}\,\hat{6}\,\hat{6}\,\hat{6} \\ \hat{6}\,\hat{6}\,\hat{\mathbf{7}} \\ \hat{6}\,\hat{7} \end{array} \quad \to \quad \begin{array}{l} \hat{7}\,\hat{7}\,\hat{7}\,\hat{7} \\ \hat{7}\,\hat{7}\,\hat{8}\,\hat{\mathbf{8}} \\ \hat{7}\,\hat{7} \\ \hat{8}\,\hat{8} \end{array}$$

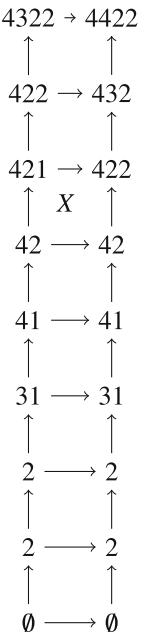
where each $\hat{i}$ denotes a box in λ^i, and each bold entry denotes an appended box. The resultant tableau is thus

$$\tilde{P} = \emptyset \prec 2 \prec 2 \prec 31 \prec 41 \prec 42 \prec 422 \prec 432 \prec 4422 = \begin{array}{l} 1\,1\,3\,4 \\ 3\,5\,7\,8 \\ 6\,6 \\ 8\,8 \end{array} .$$

One can visualise the insertion process by building the shapes up vertically.



As we can see, this forms a one-column lattice diagram such that for each $i \le k$, the vertices labelled with λ^{i-1}, λ^i, $\tilde{\lambda}^{i-1}$ and $\tilde{\lambda}^i$ surround a box, which we call the ith box. We can put an X into the kth box to indicate that the number inserted into P is k. The corresponding diagram of the previous example where we insert a 6 into tableau (1) is:



3 Symmetry property for the Robinson–Schensted algorithm

In this section, we define the insertion of words and recall Fomin's formulation of the RS algorithm in terms of growth diagrams [5–8] which we use to show the proof of the symmetry property, following the presentation given in [23, Sect. 7.13].

For a word $w = w_1 w_2 \dots w_n \in [l]^n$, the RS algorithm starts with the empty tableau $P(0) = \emptyset$. Then, w_1 is inserted into $P(0)$ to obtain $P(1)$, then w_2 into $P(1)$ to obtain $P(2)$ and so on. The recording tableau Q is a standard tableau defined by:

$$Q = \mathrm{sh}P(0) \prec \mathrm{sh}P(1) \prec \cdots \prec \mathrm{sh}P(n).$$

For example, the following table shows the process of inserting the word $w = 31342$:

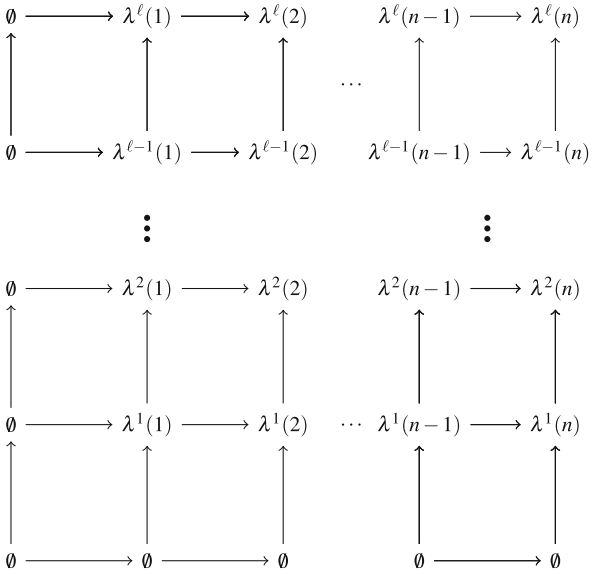
i	1	2	3	4	5
$P(i)$	3	1 3	$\begin{matrix}1&3\\3\end{matrix}$	$\begin{matrix}1&3\\3\\4\end{matrix}$	$\begin{matrix}1&3&3\\2\\4\end{matrix}$
Q^i	3	1 2	$\begin{matrix}1&2\\3\end{matrix}$	$\begin{matrix}1&2\\3\\4\end{matrix}$	$\begin{matrix}1&2&5\\3\\4\end{matrix}$

.

The corresponding pair of tableaux, which we denote as $(P(w), Q(w))$, is:

$$(P(w), Q(w)) = \begin{pmatrix} 1\,3\,3 & 1\,2\,5 \\ 2 & , \ 3 \\ 4 & 4 \end{pmatrix}$$

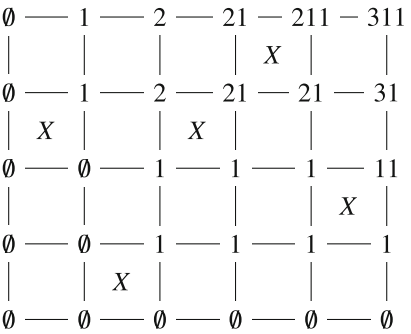
If we denote $(\lambda^k(i))_{1\le k\le \ell}$ as the shape of the subtableaux for $P(i)$, then since $P(i)$ is obtained from $P(i-1)$ by inserting w_i, we can construct a $\{0,1,\ldots,n\}\times\{0,1,\ldots,\ell\}\subset\mathbb{N}^2$ lattice growth diagram by concatenating the one-column lattice diagrams defined in the previous section. This is illustrated in the following picture.



As such, the tableau pair obtained is for P the shapes on the vertices of the rightmost vertical and for Q the shapes on the vertices on the top horizontal line:

$$P = \lambda^1(n) \nearrow \lambda^2(n) \nearrow \cdots \nearrow \lambda^\ell(n);$$
$$Q = \lambda^\ell(1) \nearrow \lambda^\ell(2) \nearrow \cdots \nearrow \lambda^\ell(n).$$

For example, if we take $\ell = 4$ for word $w = 31342$, the growth diagram is as follows:



The RS algorithm was initially defined to take a permutation as an input and output a pair of *standard* tableaux with the same shape. In this case, we take the word identified

by $\sigma(1)\sigma(2)\sigma(3)\ldots\sigma(n)$ as the input, which we also denote by σ. A classical result of the algorithm is the symmetry property:

Theorem 1 (See e.g. [9,21,23]) *For any permutation σ,*

$$(P(\sigma^{-1}), Q(\sigma^{-1})) = (Q(\sigma), P(\sigma)).$$

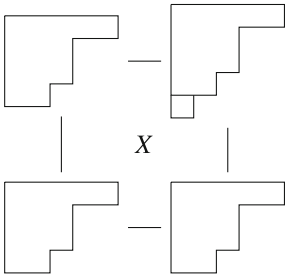
Proof (Sketch proof) We present a growth diagram proof whose row insertion counterpart can be found in [23]. Basically, the algorithm is reformulated in a way that is symmetric on the n by n growth diagram. We index a box by (m, k) if its four vertices are $(m-1, k-1)$, $(m-1, k)$, $(m, k-1)$ and (m, k). For any box (m, k), denote by $\lambda, \mu^1, \mu^2, \nu$ the partitions on $(m-1, k-1)$, $(m-1, k)$, $(m, k-1)$ and (m, k), respectively (see the diagram below).

$$\begin{array}{ccc} \mu^1 & - & \nu \\ | & & | \\ \lambda & - & \mu^2 \end{array}$$

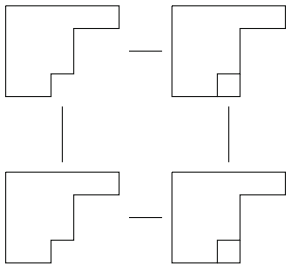
The algorithm goes with the following rule:

1. If there is an X in the box and $\lambda = \mu^1 = \mu^2$, then $\nu = \lambda + \mathbf{e}_{l(\lambda)+1}$, that is ν is obtained by adding a box that forms a new row itself at the bottom of λ.
2. If there is no X in the box and $\lambda = \mu^1$, then $\nu = \mu^2$.
3. If there is no X in the box and $\lambda = \mu^2$, then $\nu = \mu^1$.
4. If there is no X in the box and $\mu^1 = \lambda + \mathbf{e}_i$, $\mu^2 = \lambda + \mathbf{e}_j$ with $i \neq j$, then $\nu = \lambda + \mathbf{e}_i + \mathbf{e}_j = \mu^1 \cup \mu^2$.
5. If there is no X in the box and $\mu^1 = \mu^2 = \lambda + \mathbf{e}_i$, then $\nu = \lambda + \mathbf{e}_i + \mathbf{e}_{i'}$, where $i' = \max(\{j \leq i : \mu^1_{j-1} > \mu^1_j\} \cup \{1\})$.

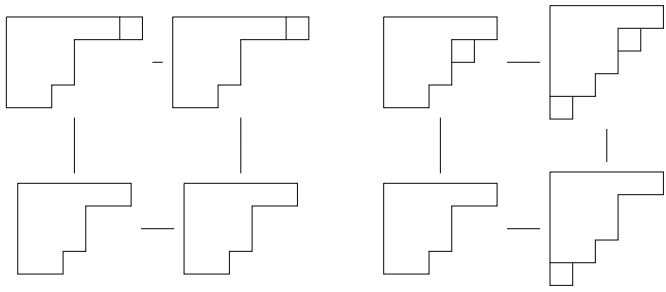
Note that these rules do not apply to the words case, which is why permutation is special. The 5 cases together with a trivial case that belongs to both Case 2 and Case 3 are illustrated as follows.



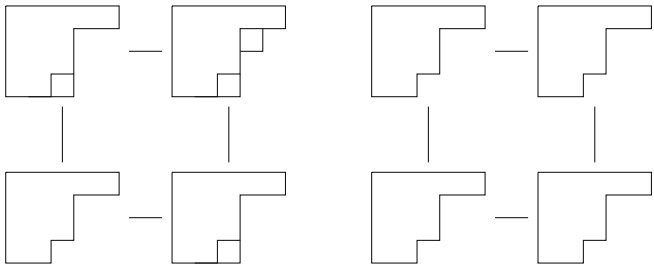
Case 1



Case 2



Case 3 Case 4



Case 5 "Boring" intersection of Case 2 and Case 3

This way, all the vertices of the diagram can be labelled recursively given $\emptyset$ as the boundary condition on the leftmost and bottom vertices. One could check that this is indeed equivalent to the definition in the previous section. Moreover, the transposition of the lattice diagram preserves the algorithm. That is: if we put Xs in boxes $(\sigma(i), i)_{i\leq n}$ rather than $(i, \sigma(i))_{i\leq n}$, and label each vertex (i, j) with what was labelled on (j, i), we end up with the configuration that is the same as if we apply the rules 1 through 5. This immediately finishes the proof.

4 A q-weighted Robinson–Schensted algorithm

In a recent paper [19], a q-weighted RS algorithm was introduced. In this and the next section, we describe the algorithm in a different way from the definition in [19]. At the end of next section, it is obvious to see that:

Proposition 2 *The algorithm described in this section for inserting a letter to a tableau and in the next section for inserting a word to an empty tableau is an equivalent reformulation of the q-weighted RS algorithm defined in [19].*

When inserting a letter to a tableau, it outputs a weighted set of tableaux. To insert a k into P, we start with λ^k, append a box to different possible rows no lower than k, record the index of rows and for each one of these indices j_k, we obtain a new kth shape $\tilde{\lambda}^k = \lambda^k + \mathbf{e}_{j_k}$ with weight $w_0(k, j_k)$; then, we proceed to add a box to all possible rows in λ^{k+1} no lower than j_k and obtain a weighted set of new $k+1$th

shapes $\{(\tilde{\lambda}^{k+1} = \lambda^{k+1} + \mathbf{e}_{j_{k+1}}, w_1(k+1, j_{k+1})) : j_{k+1} \le j_k\}$; then, for each j_{k+1}, we obtain the new $k+2$th shapes with weight $w_1(k+2, j_{k+2})$ by adding a box to j_{k+2}th row in λ^{k+2} and so on and so forth. We also prune all the 0-weighted branches.

This way we obtain a forest of trees whose roots are $\tilde{\lambda}^k$'s and leaves are $\tilde{\lambda}^\ell$'s with edges labelled by the weights. If we prepend $\emptyset \prec \lambda^1 \prec \lambda^2 \prec \cdots \prec \lambda^{k-1}$ to this forest, we obtain a tree with root $\emptyset$, the first k levels each having one branch with one edge, which we label with weight 1. Then, for each leaf $\tilde{\lambda}^\ell$, we obtain a unique tableau by reading its genealogy. We also associate this tableau with a weight which is the product of weights along the edges. By going through all leaves, we obtain a set of weighted tableaux, which is the output of q-inserting a k to P.

Now, we describe how we calculate the weights when inserting a box to ith shape λ^i with a j_{i-1} (let $j_{k-1} = k$) specified. First, let us define functions f_0 and f_1 associated with a pair of nested partitions $\mu \prec \lambda$:

$$f_0(j; \mu, \lambda) = 1 - q^{\mu_{j-1} - \lambda_j}; \qquad f_1(j; \mu, \lambda) = \frac{1 - q^{\mu_{j-1} - \lambda_j}}{1 - q^{\mu_{j-1} - \mu_j}}; \quad 1 < j \le k$$
$$f_0(1; \mu, \lambda) = f_1(1; \mu, \lambda) = 1.$$

When adding a box to the first affected partition λ^k, the weight $w_0(k, j_k)$ associated with each $j_k \le k$ is

$$w_0(k, j_k) = f_0(j_k; \lambda^{k-1}, \lambda^k) \prod_{p=j_k+1}^{k} (1 - f_0(p; \lambda^{k-1}, \lambda^k)).$$

For $r > k$, and for a fixed pair of (λ^r, j_{r-1}), when adding a box to row $j_r \le j_{r-1}$ of λ^r, the corresponding weight $w_1(r, j_r)$ is

$$w_1(r, j_r) = \begin{cases} f_0(j_r; \lambda^{r-1}, \lambda^r) \left(\prod_{p=j_r+1}^{j_{r-1}-1} (1 - f_0(p; \lambda^{r-1}, \lambda^r)) \right) & \\ \quad \times (1 - f_1(j_{r-1}; \lambda^{r-1}, \lambda^r)), & 1 \le j_r < j_{r-1} \\ f_1(j_r; \lambda^{r-1}, \lambda^r), & j_r = j_{r-1}. \end{cases}$$

For example, below is a path of the tree, i.e. genealogy of a possible output tableau when inserting a 5 into tableau (1):

$$\emptyset \xrightarrow{1} \begin{array}{|c|}\hline \hat{1} \\ \hline \end{array} \xrightarrow{1} \begin{array}{|c|c|}\hline \hat{1} & \hat{1} \\ \hline \end{array} \xrightarrow{1} \begin{array}{|c|c|c|}\hline \hat{1} & \hat{1} & \hat{1} \\ \hline \end{array} \xrightarrow{1} \begin{array}{|c|c|c|}\hline \hat{2} & \hat{2} & \hat{3} \\ \hline \hat{3} \\ \cline{1-1} \end{array} \xrightarrow{1} \begin{array}{|c|c|c|c|}\hline \hat{3} & \hat{3} & \hat{3} & \hat{4} \\ \hline \hat{3} \\ \cline{1-1} \end{array} \xrightarrow{1-q} \begin{array}{|c|c|c|c|}\hline \hat{4} & \hat{4} & \hat{4} & \hat{4} \\ \hline \hat{4} & \hat{5} \\ \cline{1-2} \hat{\mathbf{5}} \\ \cline{1-1} \end{array}$$

$$\xrightarrow{\left(1 - \frac{1-q}{1-q^2}\right)(1-q^2)} \begin{array}{|c|c|c|c|}\hline \hat{5} & \hat{5} & \hat{5} & \hat{5} \\ \hline \hat{5} & \hat{5} & \hat{\mathbf{6}} \\ \cline{1-3} \hat{6} \\ \cline{1-1} \end{array} \xrightarrow{1} \begin{array}{|c|c|c|c|}\hline \hat{6} & \hat{6} & \hat{6} & \hat{6} \\ \hline \hat{6} & \hat{6} & \hat{\mathbf{7}} \\ \cline{1-3} \hat{6} & \hat{7} \\ \cline{1-2} \end{array} \xrightarrow{\frac{1-q}{1-q^2}} \begin{array}{|c|c|c|c|}\hline \hat{7} & \hat{7} & \hat{7} & \hat{7} \\ \hline \hat{7} & \hat{7} & \hat{8} & \hat{\mathbf{8}} \\ \cline{1-4} \hat{7} & \hat{7} \\ \cline{1-2} \hat{8} & \hat{8} \\ \cline{1-2} \end{array}$$

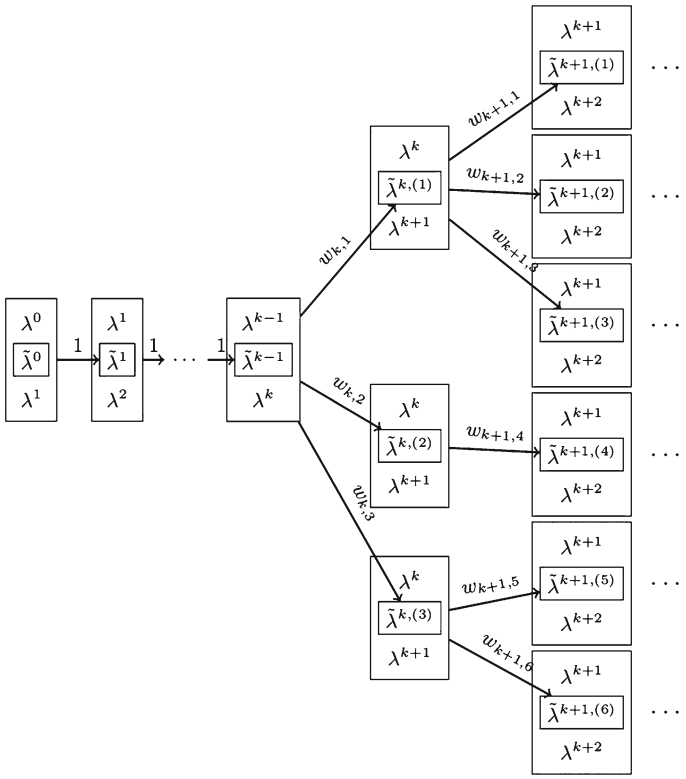


Fig. 1 The structure of a branching insertion algorithm when inserting a letter k (see Sect. 7, where the q-weighted algorithm is an example)

where again each $\hat{r}$ denotes a box in λ^r and the bold $\hat{r}$ denotes the new box $\tilde{\lambda}^r/\lambda^r$. We also have $j_5 = 3$, $j_6 = j_7 = j_8 = 2$ in this example. The above output tableau is

$$\tilde{P} = \emptyset \prec 2 \prec 2 \prec 31 \prec 41 \prec 421 \prec 431 \prec 432 \prec 4422 = \begin{array}{cccc} 1 & 1 & 3 & 4 \\ 3 & 5 & 6 & 8 \\ 5 & 7 & & \\ 8 & 8 & & \end{array}$$

with weight $1 \cdot 1 \cdot 1 \cdot 1 \cdot (1-q) \cdot \left(1 - \frac{1-q}{1-q^2}\right)(1-q^2) \cdot 1 \cdot \frac{1-q}{1-q^2} = \frac{q(1-q)^2}{1+q}$. For any two tableaux P and $\tilde{P}$, we denote the weight of obtaining $\tilde{P}$ after inserting k into P by $I_k(P, \tilde{P})$. So, in the previous example, we have $I_5(P, \tilde{P}) = \frac{q(1-q)^2}{1+q}$.

As we can see, each branching to obtain a weighted set of new ith shapes $\{\tilde{\lambda}^i\}$ is determined by λ^i and j_{i-1}, the latter in turn is identified by the pair $(\lambda^{i-1}, \tilde{\lambda}^{i-1})$. Therefore, the branching is determined by the triplet $(\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1})$. While each $\tilde{\lambda}^i$, together with λ^i and λ^{i+1}, determines a next branching. Hence, the tree has the structure illustrated in Fig. 1, where we put the weights on the edges. This way we can write theweights:

$$w((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i) = \begin{cases} \mathbb{I}_{\tilde{\lambda}^i = \lambda^i} & i < k \\ w_0(i, j_i) & i = k \\ w_1(i, j_i) & i > k \end{cases}$$

where $\tilde{\lambda}^i = \lambda^i + \mathbf{e}_{j_i}$, $j_i \leq j_{i-1}$ for $i \geq k$ and $\mathbb{I}$ is the indicator function.

Therefore, the q-insertion algorithm can be visualised in a 'one-column' graph similar to the usual insertion. Each node λ^i (or $\tilde{\lambda}^{j,(p_j)}$) representing the old ith shape (or a new jth shape) is associated with a vertex $(0, i)$ (or $(1, j)$) as the northwest (or northeast) vertex of the ith (or jth) box. Each triplet $(\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1,(p_{i-1})})$ is represented as connected triple nodes surrounding the ith box from the south and the west, for which each branch $\tilde{\lambda}^{i,(p_i)}$ is represented as a node that connects to both λ^i and $\tilde{\lambda}^{i-1,(p_{i-1})}$, where we put the weights on both the horizontal and vertical edges.

$$w(\tilde{\lambda}^{i-1,(p_{i-1})} \to \tilde{\lambda}^{i,(p_i)}) = w(\lambda^i \to \tilde{\lambda}^{i,(p_i)}) = w((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1,(p_{i-1})}), \tilde{\lambda}^i), \quad i \geq 1.$$

As is in the usual insertion case, we put an X in the kth box to indicate we are inserting a k. Figure 2 is such a visualisation corresponding to the tree in Fig. 1.

All the nodes associated with the right column $((i, 0) : 0 \leq i \leq \ell)$ form a tree, such that each $\tilde{P}$ in the output corresponds to a genealogy of a node associated with $(1, \ell)$, whose weight $I_k(P, \tilde{P})$ can be obtained as the product of the weights along the genealogical line of the tree.

5 Word input for the q-weighted Robinson–Schensted algorithm

We can also take a word as input for the q-weighted RS algorithm and produce a set of weighted pairs of tableaux. We start with the set of only one pair of tableaux $(P(0), Q^0) = (\emptyset, \emptyset)$ with weight 1. Suppose at time m, we have obtained a set of weighted pairs of tableaux

$$\{(P(m)^{(1)}, (Q^m)^{(1)}, w(m)^{(1)}), \ldots, (P(m)^{(p)}, (Q^m)^{(p)}, w(m)^{(p)})\}.$$

We use brackets in superscripts to indicate the different possibilities, in order not to confuse with subtableaux or shapes. We take each triplet $(P(m)^{(i)}, (Q^m)^{(i)}, w(m)^{(i)})$ in the collection and insert w_{m+1} into $P(m)^{(i)}$ to produce a new set of P-tableaux paired with the weight generated during the insertion

$$\{(P(m+1)^{(i,1)}, w^{(i,1)}), (P(m+1)^{(i,2)}, w^{(i,2)}), \ldots, (P(m+1)^{(i,r_i)}, w^{(i,r_i)})\}.$$

Then, for each $P(m+1)^{(i,j)}$, $1 \leq j \leq r_i$, we pair it with a $(Q^{m+1})^{(i,j)}$ by adding a box with entry $m+1$ to $(Q^m)^{(i)}$ such that $\mathrm{sh}P(m+1)^{(i,j)} = \mathrm{sh}(Q^{m+1})^{(i,j)}$. We also attach a weight $w(m+1)^{(i,j)} = w(m)w^{(i,j)}$ to this pair. This way, for each triplet

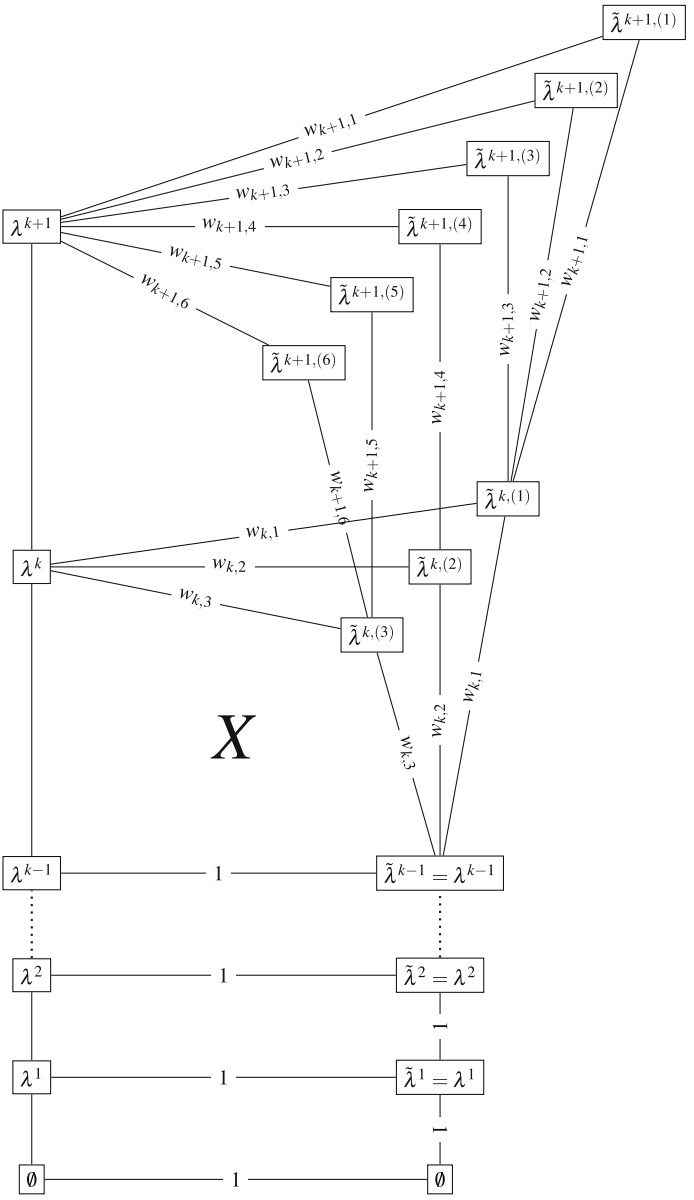


Fig. 2 The one-column construction of q-inserting a letter to a tableau

$(P(m)^{(i)}, (Q^m)^{(i)}, w(m)^{(i)})$, we have obtained its branching set

$$\{(P(m+1)^{(i,1)}, (Q^{m+1})^{(i,1)}, w(m+1)^{(i,1)}),$$
$$(P(m+1)^{(i,2)}, (Q^{m+1})^{(i,2)}, w(m+1)^{(i,2)}),$$
$$\ldots, (P(m+1)^{(i,r_i)}, (Q^{m+1})^{(i,r_i)}, w(m+1)^{(i,r_i)})\}.$$

Let i run over $1, 2, \ldots, p$; we obtained a collection of all possible branchings:

$$((P(m+1)^{(i,j)}, (Q^{m+1})^{(i,j)}, w(m+1)^{(i,j)}))_{1\le i\le p, 1\le j\le r_i}.$$

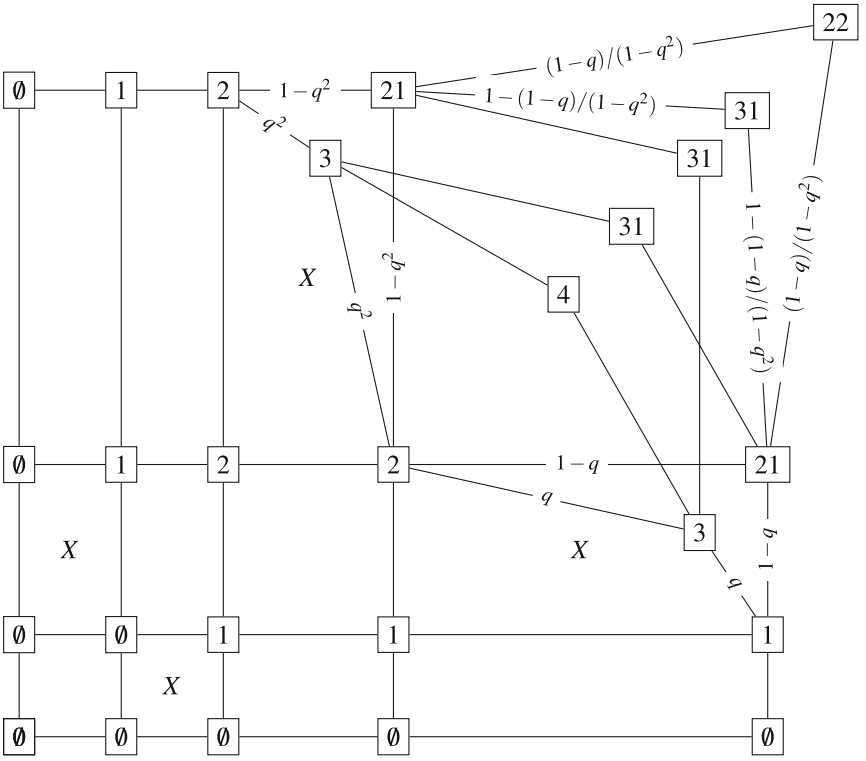
Note by *collection* we mean a vector of objects that allows repeats, as opposed to a *set*. We merge triplets with the same tableau pair by adding up their weights. This way, we have obtained the weighted set of pairs of tableaux at time $m+1$.

The output of inserting the word $w_1 w_2 \ldots w_n$ is defined as the weighted set at time n. We denote by $\phi_w(P, Q)$ the weight of pair (P, Q) in this set. This is defined recursively by the following formula (see [19]). For P and Q with the same shape of size n and word w of length $n-1$,

$$\phi_{wk}(P, Q) = \sum \phi_w(\hat{P}, Q^{n-1}) I_k(\hat{P}, P),$$

where the sum is over all tableau $\hat{P}$ with the same shape as Q^{n-1}.

This, like in the usual case, can be visualised as a graph whose nodes are associated with vertices on $D_{n,\ell} := \{0, \ldots, n\} \times \{0, \ldots, \ell\}$ lattice diagram by concatenating the graphs of the one-column construction in Fig. 2 associated with the insertion of each letter. Each possible $\lambda^{k,(p_{m,k})}(m)$ is associated with the vertex (m, k). For example, we obtain the following graph if we apply the algorithm to $2132 \in [3]^4$, where the unlabelled edges have weight 1, which will be the case hereafter.



From the definition of the algorithm, each node $\lambda^{\ell,(p)}(n)$ at (n, ℓ) corresponds to a (P, Q) tableau pair in the output set. Indeed, we can trace back the genealogy of $\lambda^{\ell,(p)}(n)$ and find a unique array of indices $(p_{m,k})_{0\le m\le n, 0\le k\le \ell}$ with $p_{n,\ell} = p$ such that nodes $(\lambda^{k,(p_{m,k})}(m))_{m,k}$ are connected and form a diagram that is isomorphic to the lattice diagram $D_{n,\ell}$. Clearly,

$$P^{(p)} = \lambda^{0,(p_{n,0})}(n) \prec \lambda^{1,(p_{n,1})}(n) \prec \cdots \prec \lambda^{\ell,(p_{n,\ell})}(n);$$
$$Q^{(p)} = \lambda^{\ell,(p_{0,\ell})}(0) \prec \lambda^{\ell,(p_{1,\ell})}(1) \prec \cdots \prec \lambda^{\ell,(p_{n,\ell})}(n),$$

are the corresponding (P, Q) tableau pair. Moreover, we can identify a weight with the node $\lambda^{\ell,(p)}$ by multiplying the weights of all the horizontal (or all the vertical) edges along its genealogical diagram and denote it by $w(\lambda^{\ell,(p)})$:

$$\begin{aligned} w(\lambda^{\ell,(p)}(n)) &= \prod_{1\le m\le n, 0\le k\le \ell} w(\lambda^{k,(p_{m-1,k})}(m-1) \to \lambda^{k,(p_{m,k})}(m)) \\ &= \prod_{0\le m\le n, 1\le k\le \ell} w(\lambda^{k-1,(p_{m,k-1})}(m) \to \lambda^{k,(p_{m,k-1})}(m)). \end{aligned}$$

Then, by definition,

$$\phi_w(P, Q) = \sum w(\lambda^{\ell,(p)}(n))\mathbb{I}_{P^{(p)}=P, Q^{(p)}=Q},$$

where the sum is over all nodes $\lambda^{l,(p)}(n)$ at vertex (n, l).

Note that when $q \in (0, 1)$, for a fixed word w, the weights are all non-negative and have a total sum 1:

$$\phi_w(P, Q) \ge 0, \qquad \sum \phi_w(P, Q) = 1,$$

where the sum is over all pairs of semi-standard and standard tableau pairs (P, Q). This means the algorithm can be viewed as a randomised version of the usual RS algorithm, see [19].

6 The symmetry property for the q-weighted RS algorithm with permutation input

When we take a permutation input σ which is identified as a word with distinct letters in the same way as in the usual RS algorithm, we also end up with a symmetry property, which is the main result of this paper:

Theorem 3 *For any permutation $\sigma \in S_n$ and standard tableau pair (P, Q),*

$$\phi_{\sigma^{-1}}(Q, P) = \phi_\sigma(P, Q).$$

As in the usual RS algorithm case, permutation input is special in that we can restate the insertion rule in a symmetric fashion. For each box, we pick a connected

triplet on southwest, northwest and southeast corners. That is, suppose the box has index (m, k) we fix arbitrary p_1, p_2 and p_3 such that $(\lambda, \mu^1, \mu^2) := (\lambda^{k-1,(p_1)}(m-1), \lambda^{k,(p_2)}(m-1), \lambda^{k-1,(p_3)}(m))$ is a connected triplet. We denote by N the set of partitions on the vertex (m, k) that are connected to μ^1 and μ^2, and for any $\nu \in N$, we write $w(\mu, \nu)$ instead of $w(\mu^1 \to \nu)$ or $w(\mu^2 \to \nu)$ since they are equal, and for the sake of symmetry. Define an operator $I^k : W \to W$ by

$$I^k \lambda := \lambda + \mathbf{e}_{\max\{j \le k : \lambda + \mathbf{e}_j \in W\}}$$

and a set $\Lambda^k(\lambda) \subset W$ by

$$\Lambda^k(\lambda) := \{I^j \lambda : j \le k\},$$

and denote $\Lambda(\lambda) := \Lambda^{l(\lambda)+1}(\lambda)$.

Then, N and $(w(\mu, \nu) : \nu \in N)$ belong to one of the following 5 cases.

1. The box has an X in it, and μ^1, μ^2 are equal to λ. Then, N consists of all possible partitions obtained by adding a box to λ:

$$N = \Lambda(\lambda).$$

The weights are

$$w(\mu, \nu) = \begin{cases} q^{\lambda_j} - q^{\lambda_{j-1}}, & \text{if } \nu = \lambda + \mathbf{e}_j \text{ for some } j > 1 \\ q^{\lambda_1}, & \text{if } \nu = \lambda + \mathbf{e}_1 \end{cases}, \quad \nu \in N.$$

2. The box does not have an X in it and $\mu^1 = \lambda$. Then, N is a singleton which is the same as μ^2, and the weight is 1:

$$w(\mu, \nu) = 1, \quad \nu \in N = \{\mu^2\}.$$

3. (The dual case of case 2) The box does not have an X in it and $\mu^2 = \lambda$. Then, N is a singleton which is the same as μ^1, and the weight is 1:

$$w(\mu, \nu) = 1, \quad \nu \in N = \{\mu^1\}.$$

4. The box is empty. $\mu^1 = \lambda + \mathbf{e}_i$ and $\mu^2 = \lambda + \mathbf{e}_j$ for some $i \neq j$. Then, N again only contains one element $\mu^1 \cup \mu^2$, with weight 1:

$$w(\mu, \nu) = 1, \quad \nu \in N = \{\mu^1 \cup \mu^2\}.$$

5. The box is empty and $\mu^1 = \mu^2 = \lambda + \mathbf{e}_i := \mu$ for some i. Then, N consists of all possible partitions that are obtained by adding a box to a row no lower than ith row of μ. That is

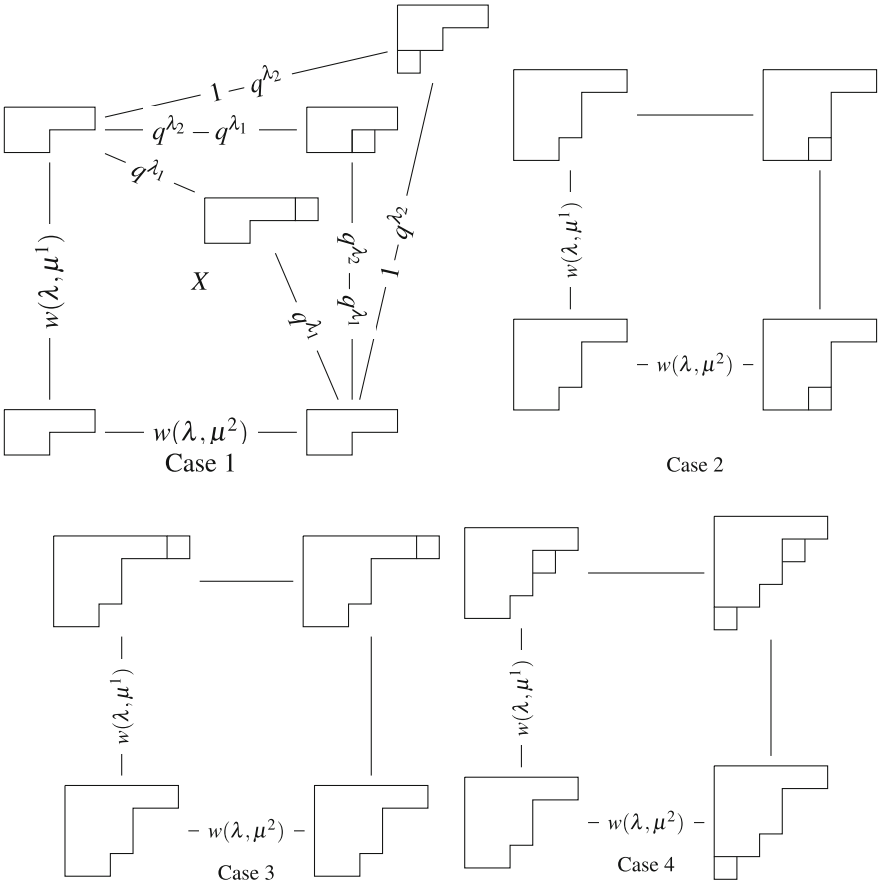
$$N = \Lambda^i(\mu).$$

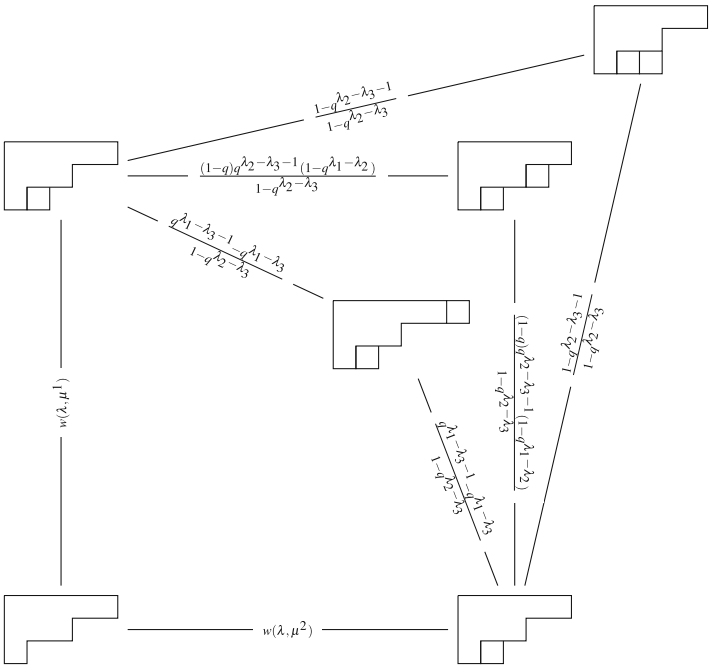
The weights are

$$w(\mu,\nu) = \begin{cases} \frac{1-q^{\lambda_{i-1}-\lambda_i-1}}{1-q^{\lambda_{i-1}-\lambda_i}}, & \nu=\mu+\mathbf{e}_i; \\ \frac{1-q}{1-q^{\lambda_{i-1}-\lambda_i}} q^{\lambda_j-\lambda_i-1}(1-q^{\lambda_{j-1}-\lambda_j}), & \nu=\mu+\mathbf{e}_j \text{ for some } 2\le j<i; \\ \frac{1-q}{1-q^{\lambda_{i-1}-\lambda_i}} q^{\lambda_1-\lambda_i-1}, & \nu=\mu+\mathbf{e}_1. \end{cases}$$

$$, \nu \in N.$$

We call this the *growth graph rule* as opposed to the *insertion rule* described in Sect. 4 and 5. Below is a group of illustrations of all 5 cases, where λ is the southwest partition.





Case 5

Proof (Proof of Theorem 3) It now suffices to show that the above construction agrees with the definition of the algorithm in the preceding section. That is, for any connected triplets (λ, μ^1, μ^2) surrounding the box (m, k) from the southwest, its branching set with corresponding weights according to the insertion rule agrees with N and $w(\mu, \nu)$'s according to the growth graph rule. To distinguish the context of insertion rule and growth graph rule, we also write $\lambda = \lambda^{k-1}$, $\mu^1 = \lambda^k$, $\mu^2 = \tilde{\lambda}^{k-1}$ and $\nu = \tilde{\lambda}^k \in N$ in accordance with the definition of the insertion algorithm in Sect. 4. We show this by discussing the location of (m, k) in the permutation matrix of σ, which corresponds to the 5 cases in the growth graph rule.

1. The box (m, k) has an X in it. This means $\sigma_m = k$. So, we are inserting a k to the tableau at time m and hence $\lambda^{k-1} = \tilde{\lambda}^{k-1}$, i.e. $\lambda = \mu^2$. Moreover, since σ is a permutation, we have $\sigma_i \neq k$ for all $i < m$, hence $\lambda^{k-1} = \lambda^k$, i.e. $\lambda = \mu^1$ (condition of Case 1 satisfied). Any branch of $(\lambda^{k-1}, \lambda^k, \tilde{\lambda}^{k-1}) = (\lambda, \lambda, \lambda)$ is one of the $\lambda + \mathbf{e}_j$'s such that the weight $w((\lambda, \lambda, \lambda), \lambda + \mathbf{e}_j) = w_0(k, j) \neq 0$.

$$w_0(k, j) = f_0(j; \lambda, \lambda) \prod_{p=j+1}^{k} (1 - f_0(p; \lambda, \lambda))$$
$$= \begin{cases} (1 - q^{\lambda_{j-1} - \lambda_j}) q^{\lambda_j} & \text{if } j > 1 \\ q^{\lambda_1} & \text{if } j = 1. \end{cases}$$

So, the weights agree. All the pruned branches $\lambda + \mathbf{e}_j$ with $w_0(k, j) = 0$ are exactly the $\lambda + \mathbf{e}_j$'s that are not partitions. Therefore, N is exactly the set of all branches of the triplet.

2. There is no X in (i, k) for $1 \le i \le m$. This means $\sigma_i \ne k$ for $i \le m$. Since $\sigma_i \ne k$ for $i < m$, $\lambda^{k-1} = \lambda^k$, i.e. $\lambda = \mu^1$ (condition of Case 2 satisfied). Moreover, since $\sigma_m \ne k$, the triplet produces only one branch which is equal to μ^2 with weight 1. This agrees with Case 2.
3. There is no X in (m, i) for $1 \le i \le k$. This means $\sigma_m > k$. By the insertion rule, $\lambda^{k-1} = \tilde{\lambda}^{k-1}$, i.e. $\lambda = \mu^2$ (condition of Case 3 satisfied). Again, by the same rule, the triplet only produces one branch that equals μ^1 with weight 1.
4. There is one X in each of (t, k) and (m, s) for some $t < m$ and $s < k$. Then, on the one hand, $\sigma_t = k$ so $\lambda^k = \lambda^{k-1} + \mathbf{e}_i$ for some i, i.e. $\mu^1 = \lambda + \mathbf{e}_i$. On the other hand, $\sigma_m < k$ so $\tilde{\lambda}^{k-1} = \lambda^{k-1} + \mathbf{e}_j$ for some j and $j_{k-1} = j$, i.e. $\mu^2 = \lambda + \mathbf{e}_j$ (condition of Case 4 or Case 5 satisfied). Therefore, the branches of the triplet $(\lambda, \lambda + \mathbf{e}_i, \lambda + \mathbf{e}_j)$ are in the form of $\tilde{\lambda}^k = \lambda^k + \mathbf{e}_{j_k}$ for $j_k \le j$ with weights

$$w((\lambda, \lambda + \mathbf{e}_i, \lambda + \mathbf{e}_j), \lambda + \mathbf{e}_i + \mathbf{e}_{j_k}) = w_1(k, j_k)$$
$$= \begin{cases} f_0(j_k; \lambda, \lambda + \mathbf{e}_i) \left(\prod_{p=j_k+1}^{j-1} (1 - f_0(p; \lambda, \lambda + \mathbf{e}_i)) \right) & \\ \times (1 - f_1(j; \lambda, \lambda + \mathbf{e}_i)) & \text{if } j_k < j \\ f_1(j; \lambda, \lambda + \mathbf{e}_i) & \text{if } j_k = j \end{cases} \tag{2}$$

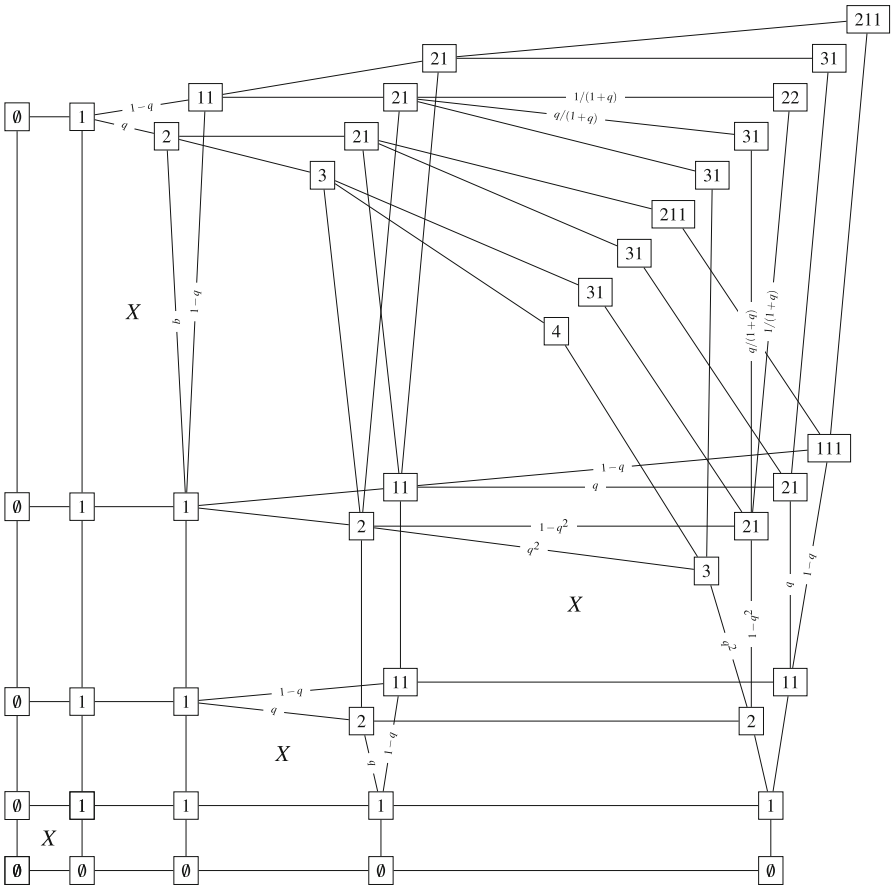
If $j \ne i$ (condition of Case 4 satisfied), then $f_1(j; \lambda, \lambda + \mathbf{e}_i) = (1 - q^{\lambda_{j-1} - \lambda_j})/(1 - q^{\lambda_{j-1} - \lambda_j}) = 1$. Therefore, the branch has only one shape equal to $\mu^1 + \mathbf{e}_j$ with weight 1. This agrees with Case 4.
If $j = i$ (condition of Case 5 satisfied), then by (2),

$$w_1(k, j_k) = \begin{cases} \frac{1 - q^{\lambda_{i-1} - \lambda_i - 1}}{1 - q^{\lambda_{i-1} - \lambda_i}}; & \text{if } j_k = i; \\ \frac{1 - q}{1 - q^{\lambda_{i-1} - \lambda_i}} q^{\lambda_j - \lambda_i - 1} (1 - q^{\lambda_{j-1} - \lambda_j}); & \text{if } 2 \le j_k < i; \\ \frac{1 - q}{1 - q^{\lambda_{i-1} - \lambda_i}} q^{\lambda_1 - \lambda_i - 1}; & \text{if } j_k = 1. \end{cases}$$

So, the weights agree with Case 5. Moreover, all the pruned branches are exactly the $\lambda + \mathbf{e}_i + \mathbf{e}_{j_k}$'s that are not partitions. Therefore, N is indeed the set of all branches.

When inverting a permutation σ to σ^{-1}, the X marks are transposed, so is the weighted graph by the symmetry of the rule, and thus, we have arrived at the conclusion.

Note that in both rules although we consider an arbitrary box with index (m, k), the rules do not depend on either m or k. This is a key condition for the symmetry property to work and will be generalised in Proposition 6 in the following section. Below is the growth graph of the permutation 1423:



The growth graphs defined above are a natural q-generalisation of the growth diagrams introduced by Fomin [5–8] and reduce to Fomin's growth diagrams when $q \to 0$.

7 More insertion algorithms

In [23], it was explained how to use Fomin's growth diagram technique to show the symmetry property for the RS algorithm with *row insertion*. To row insert a k into a tableau P, we again keep $\lambda^0 \prec \lambda^1 \prec \cdots \prec \lambda^{k-1}$ unchanged. Then, we append a box at the end of first row of $\lambda^k, \lambda^{k+1}, \ldots, \lambda^{k_1-1}$, where k_1 is the smallest number in row 1 of P that is larger than k, then we append a box at the end of second row of $\lambda^{k_1}, \lambda^{k_1+1}, \ldots, \lambda^{k_2-1}$, where k_2 is the smallest number in row 2 of P that is larger than k_1 and so on and so forth. More precisely, define:

$$k_0 = k; \quad k_j = \min(\{k' > k_{j-1} : \lambda_j^{k'} > \lambda_j^{k'-1}\} \cup \{\ell + 1\}), \quad j \geq 1.$$

Then, the output tableau has:

$$\tilde{\lambda}^i = \begin{cases} \lambda^i, & \text{if } i < k, \\ \lambda^i + \mathbf{e}_j, & \text{if } k_{j-1} \le i < k_j \text{ for some } j. \end{cases}$$

For example, if we insert a 3 into the tableau (1), the one-column insertion diagram is as follows:

$$\begin{array}{ccc}
4322 & \rightarrow & 43221 \\
\uparrow & & \uparrow \\
422 & \rightarrow & 4221 \\
\uparrow & & \uparrow \\
421 & \rightarrow & 4211 \\
\uparrow & & \uparrow \\
42 & \longrightarrow & 421 \\
\uparrow & & \uparrow \\
41 & \longrightarrow & 42 \\
\uparrow & & \uparrow \\
31 & \longrightarrow & 41 \\
\uparrow & X & \uparrow \\
2 & \longrightarrow & 2 \\
\uparrow & & \uparrow \\
2 & \longrightarrow & 2 \\
\uparrow & & \uparrow \\
\emptyset & \longrightarrow & \emptyset
\end{array}$$

and the corresponding output tableau is

$$\tilde{P} = \begin{array}{llll} 1 & 1 & 3 & 3 \\ 3 & 4 & 8 & \\ 5 & 7 & & \\ 6 & 8 & & \\ 8 & & & \end{array} .$$

The row insertion of a word is defined in the same way as in column inserting a word. The usual row and column insertions are related in a few ways.

The first is a well-known algebraic duality relation, which we now recall. For any tableau T and positive integers x, y, denote by c_x (resp. r_y) the operation of column (resp. row) inserting x (resp. y), that is, $c_x T$ (resp. $r_y T$) is the output tableau when column (resp. row) inserting x (resp. y) into T. Denote by $(P_{\mathrm{col}}(w), Q_{\mathrm{col}}(w))$ (resp. $(P_{\mathrm{row}}(w), Q_{\mathrm{row}}(w))$) the tableau pairs when applying column (resp. row) insertions to word w, respectively. Thus, we have $P_{\mathrm{col}}(w) = c_{w_n} c_{w_{n-1}} \ldots c_{w_1} \emptyset$ and $P_{\mathrm{row}}(w) = r_{w_n} r_{w_{n-1}} \ldots r_{w_1} \emptyset$. Also, denote by w^r the inverse word of w:

$w^r = (w_n, w_{n-1}, \ldots, w_1)$. Moreover, denote by $\mathrm{ev}(Q)$ the evacuation operation on Q, see e.g. [9,21]. Then,

Proposition 4 [see e.g. [9]] *For any tableau T and integers x, y, the column insertion of x and the column insertion of y commute:*

$$r_y c_x T = c_x r_y T. \tag{3}$$

This implies for any word w,

$$P_{col}(w) = P_{row}(w^r). \tag{4}$$

If furthermore w is a permutation, then

$$Q_{col}(w) = (ev(Q_{row}(w^r)))'$$

where T' means the transposition of tableau T.

This duality has a matrix input generalisation where $Q_{\mathrm{row}}(w^r)$ is obtained by a reverse sliding operation, see e.g. [9].

Another simple relation is between the original definitions. Row insertion was initially defined as an algorithm of inserting and bumping based on the ordering of integers. This definition turns into column insertion if one replaces all occurrence of 'row' by 'column' and replaces the strong order (greater than) with the weak order (greater than or equal to) due to the asymmetry of the ordering in rows and columns in the definition of a tableau. For these definitions, see e.g. [21].

There is also a third relation which is less well-known and can be seen if one describes the 'dynamics' of the column and row insertion algorithms in terms of the (Gelfand–Tsetlin) coordinates $(\lambda_j^k)_{1\le j\le k\le \ell}$, where λ^k is the kth shape of the P-tableau. In column insertion, we initialise $j_{k-1} = j$. Then, in each step, we find the largest row index $j_k \le j_{k-1}$ for a letter k such that $\lambda_{j_{k-1}}^{k-1} > \lambda_{j_k}^k$, append a box to $\lambda_{j_k}^k$ and increase k by 1. In row insertion, we initialise $k_0 = k$, and in each step one we find the smallest letter $k_j > k_{j-1}$ for a row index j such that $\lambda_j^{k_j} > \lambda_j^{k_j-1}$, append a box to $\lambda_j^{k_{j-1}}, \ldots, \lambda_j^{k_j-1}$ and increase j by 1. We shall refer to this relation as the *dynamical duality*.

In a recent paper [2], a q-weighted RS algorithm with row insertion was proposed. It is defined in a similar way as the q-column insertion and is related to the latter by an analogue of the dynamical duality described above.

In q-column insertion, we initialise $j_{k-1} = k$. Then, in each step, we run over all row indices $j_k \le j_{k-1}$ for a letter k, with some weight append a box to $\lambda_{j_k}^k$ and increase k by 1. In q-row insertion, we initialise $k_0 = k$. In each step one, we run over all letters $k_j > k_{j-1}$ for a row index j such that, with some weight, we append a box to $\lambda_j^{k_{j-1}}, \ldots, \lambda_j^{k_j-1}$ and increase j by 1.

Our definition here is a reformulation equivalent to the one in [2]. For $\mu \prec \lambda$, define

$$g(1;\mu,\lambda) = 1 - q^{\lambda_1-\mu_1},$$
$$g(j;\mu,\lambda) = \frac{1-q^{\lambda_j-\mu_j}}{1-q^{\mu_{j-1}-\mu_j}} \quad j \geq 2.$$

When q-row inserting a k into a tableau P, we keep the first $k-1$ shapes unchanged: $\tilde{\lambda}^i = \lambda^i$, $i = 1, \ldots, k-1$, with weight 1, $\tilde{\lambda}^k = \lambda + \mathbf{e}_1$ and for $i > k$, we have a binary branching for each triplet $(\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1} = \lambda^{i-1} + \mathbf{e}_{j_{i-1}})$:

$$u((\lambda^{i-1}, \lambda^i, \lambda^{i-1} + \mathbf{e}_{j_{i-1}}), \lambda^i + \mathbf{e}_{j_i}) = \begin{cases} g(j_{i-1}; \lambda^{i-1}, \lambda^i), & j_i = j_{i-1} + 1; \\ 1 - g(j_{i-1}; \lambda^{i-1}, \lambda^i), & j_i = j_{i-1}; \\ 0, & \text{otherwise.} \end{cases}$$

Denote by $J_k(P, \tilde{P})$ the weight of obtaining a $\tilde{P}$ after row inserting a k into P.

We define the q-row insertion for word input in the same way as q-column insertion of words: successively q-row inserting letters, multiplying the weights, keeping a tableau Q to record changes for each P and merging the same tableau pairs by adding up the weights. Denote by $\psi_w(P, Q)$ the weight of obtaining tableau pair (P, Q) after q-row inserting a word w. Then, we have the same recursion rule: for a pair of tableaux (P, Q) with the same shape of size n and a word w of length $n-1$,

$$\psi_{wk}(P, Q) = \sum \psi_w(\hat{P}, Q^{n-1}) J_k(\hat{P}, P).$$

where the sum is over all tableau $\hat{P}$ with the same shape as Q^{n-1}. Since the algorithm has a similar triplet branching structure as in Fig. 1, we can build the one-column insertion construction like in Fig. 2 and concatenate them into a growth graph. With the same approach we can show the symmetry property for this algorithm.

Theorem 5 *For any permutation $\sigma \in S_n$ and standard tableau pair (P, Q),*

$$\psi_{\sigma^{-1}}(Q, P) = \psi_\sigma(P, Q).$$

Proof (A sketch proof) Again, denote by λ, μ^1, μ^2 nodes associated with vertices surrounding a box from the south and the west. The set M of all partitions branched from the triplet with $(u(\mu, \nu) : \nu \in M)$ belongs to one of the following 5 cases.

1. The box has an X in it, and μ^1, μ^2 are equal to λ. Then, M consists of only one partition $\lambda + \mathbf{e}_1$ with weight 1.
2. The box does not have an X in it, and $\mu^1 = \lambda$. Then, M consists of only one partition μ^2 with weight 1.
3. The box does not have an X in it, and $\mu^2 = \lambda$. Then, M consists of only one partition μ^1 with weight 1.
4. The box is empty. $\mu^1 = \lambda + \mathbf{e}_i$ and $\mu^2 = \lambda + \mathbf{e}_j$ for some $i \neq j$. Then, M again only contains one element $\mu^1 \cup \mu^2$ with weight 1.

5. The box is empty and $\mu^1 = \mu^2 = \lambda + \mathbf{e}_i := \mu$ for some i. Then, $M = \{\lambda + \mathbf{e}_i + \mathbf{e}_{i+1}, \lambda + 2\mathbf{e}_i\}$ with weight

$$u((\lambda, \lambda + \mathbf{e}_i, \lambda + \mathbf{e}_i), \lambda + \mathbf{e}_{i+1}) = \begin{cases} 1 - q, & \text{if } i = 1; \\ \frac{1-q}{1-q^{\lambda_{i-1}-\lambda_i}}, & \text{if } i > 1. \end{cases}$$

$$u((\lambda, \lambda + \mathbf{e}_i, \lambda + \mathbf{e}_i), \lambda + 2\mathbf{e}_i) = \begin{cases} q, & \text{if } i = 1; \\ 1 - \frac{1-q}{1-q^{\lambda_{i-1}-\lambda_i}}, & \text{if } i > 1. \end{cases}$$

The claim is concluded once these symmetric rules are verified to be equivalent to the insertion rule, which is done in the same way as in the proof of Theorem 3.

As a remark, let us point out that there is no direct generalisation of the algebraic duality relation between the column and row insertions as in Proposition 4 to the q-weighted case. More specifically, the distribution of the P-tableaux of q-column inserting a word w is not necessarily the same as that of q-row inserting the reversed word w^r, even when w is a permutation. That is, if for a standard tableau P and a permutation σ we define

$$\Phi_\sigma(P) := \sum_{Q \in \mathcal{S}} \phi_\sigma(P, Q);$$
$$\Psi_\sigma(P) := \sum_{Q \in \mathcal{S}} \psi_\sigma(P, Q),$$

then there exist a positive integer n, a permutation $\sigma \in S_n$ and a standard tableau $P \in \mathcal{S}_n$ such that

$$\Phi_\sigma(P) \neq \Psi_{\sigma^r}(P).$$

For example, taking $n = 4$, $\sigma = 2143$ and $P = \begin{matrix} 1 & 2 \\ 3 & \\ 4 & \end{matrix}$, there is no P in the output P-tableaux of q-column inserting σ, i.e. $\Phi_\sigma(P) = 0$, but the q-row insertion of the reversed permutation word does produce P, with the weight $\Psi_{\sigma^r}(P) = q(1-q)^2/(1+q)$. On the other hand, since the commutation relation (3) of the column and row insertion algorithms implies the P-duality (4), which can be carried over to the q-case, by contraposition, the q-column and q-row insertions do not commute. More precisely, for letters x and y and tableaux P and $\tilde{P}$, define by $IJ_{xy}(P, \tilde{P})$ the weight of obtaining $\tilde{P}$ by first q-column inserting x then q-row inserting y, and by $JI_{yx}(P, \tilde{P})$ the weight by first q-row inserting y then q-column inserting x:

$$IJ_{xy} := \sum_{\hat{P}} I_x(P, \hat{P}) J_y(\hat{P}, \tilde{P});$$
$$JI_{yx} := \sum_{\hat{P}} J_y(P, \hat{P}) I_x(\hat{P}, \tilde{P}),$$

then there exist two positive integers x, y and tableaux P, $\tilde{P}$ such that

$$IJ_{xy}(P, \tilde{P}) \neq JI_{yx}(P, \tilde{P}).$$

As an example, let $x = 3$, $y = 2$, $P = \begin{matrix} 1 \\ 4 \end{matrix}$ and $\tilde{P} = \begin{matrix} 1 & 2 \\ 3 \\ 4 \end{matrix}$, then $IJ_{32}(P, \tilde{P}) = q(1-q)/(1+q)$ but $JI_{23}(P, \tilde{P}) = 0$.

Let us go back to the symmetry property. More generally, the symmetry property does not require very strong conditions on the insertion algorithms. Here, we give a sufficient condition for an insertion algorithm to have this property.

First, we define a *branching insertion algorithm* as an algorithm that has the branching structure as in Fig. 1 when we insert a letter to a tableau. That is, there exists an *initial branching weight function* w_0, a *high level weight function* w_1 and a *low level weight function* w_2 such that when inserting a letter k, the weight of a new ith shape is:

$$w((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i) = \begin{cases} w_0((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i), & \text{if } i = k; \\ w_1((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i), & \text{if } i > k; \\ w_2((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i), & \text{if } i < k. \end{cases}$$

Then, we have

Proposition 6 *If a branching insertion algorithm satisfies the following conditions:*

(i) *The insertion of k' into a tableau results in increment of one coordinate by 1 in $\lambda^{k'}, \lambda^{k'+1}, \ldots, \lambda^{\ell}$, while keep $\lambda^0, \lambda^1, \ldots, \lambda^{k'-1}$ unchanged, that is, the support of $w_0((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i)$ and $w_1((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i)$ are in $\{\lambda^i + \mathbf{e}_j : j \geq 1\}$ and $w_2((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i) = \mathbb{I}_{\lambda^i = \tilde{\lambda}^i}$,*

(ii) *$w_1((\lambda^{m-1}, \lambda^m, \lambda^{m-1} + \mathbf{e}_i), \lambda^m + \mathbf{e}_j) = \mathbb{I}_{i=j}$ if $\lambda^m_i = \lambda^{m-1}_i$;*

(iii) *$w_0((\lambda, \lambda, \lambda), \lambda + \mathbf{e}_i)$ does not depend on the inserted letter;*

(iv) *$w_1((\lambda, \lambda + \mathbf{e}_i, \lambda + \mathbf{e}_i), \lambda + \mathbf{e}_i + \mathbf{e}_j)$ does not depend on the inserted letter,*

then it has the symmetry property.

Proof (Sketch proof) We use the same notations λ, μ^1, μ^2 as in descriptions of growth graph rules such that they surround the box (m, k) from the south and the west. The symmetry property is shown once we can construct a symmetric growth graph rule, where the relation of λ, μ^1 and μ^2 and whether there is an X in the box, determines the location of the box in the permutation matrix, and vice versa.

Then, the 5 cases of the growth diagram rule are satisfied given these conditions: for Case 1, the equivalence between ($\lambda = \mu^1 = \mu^2$ AND X) and ($\sigma_m = k$) is given by (i) and (ii), and the branched shapes and weights are given by (iii); for Case 2, both the equivalence between ($\lambda = \mu^1$ AND NOT X) and ($\sigma_s \neq k \forall s \leq m$) is given by (ii), and the singleton branching with weight 1 is also given by (ii); for Case 3, both the equivalence between ($\lambda = \mu^2$ AND NOT X) and ($\sigma_m > k$) is given by (i), and the singleton shape with weight 1 is also given by (i); for Case 4 and

5, the equivalence between ($\mu^1 = \lambda + \mathbf{e}_i$ AND $\mu^2 = \lambda + \mathbf{e}_j$ AND NOT X) and ($\sigma_k^{-1} < m$ AND $\sigma_m < k$) is given by (i) (ii), and the singleton shape with weight 1 in Case 4 is given by (ii) while the branched shapes and weights in Case 5 are given by (iv).

With this proposition, we can test different algorithms for the symmetry property. For example, a family of $n!$ q-weighted algorithms is defined in [2], see Dynamics 3 in Sect. 6.5.3 and (8.5) in Sect. 8.2.1 in that paper. They are indexed by vectors $h = (h_1, h_2, \ldots, h_n)$ with h_i being a positive integer less than or equal to i. They are branching algorithms. The algorithm indexed by $h = (1, 1, \ldots, 1)$ is the q-weighted row insertion, and as we have seen in Theorem 5 it has the symmetry property.

The algorithm indexed by $h = (1, 2, \ldots, n)$ also satisfies the conditions in Proposition 6 and, therefore, has a symmetry property. For convenience, we refer to this algorithm as $A_{1:n}$ in the following. Here, we restate the definition of $A_{1:n}$ in the framework of branching insertion algorithms. Denote for $\mu \prec \lambda$:

$$I^j(\lambda; \mu) := \lambda + \mathbf{e}_{\max(\{i \le j : \mu_{i-1} > \lambda_i\} \cup \{1\})}.$$

The algorithm is defined by the following weights

$$\begin{aligned}
&w_0((\lambda^{k-1}, \lambda^k, \tilde{\lambda}^{k-1}), \tilde{\lambda}^k) = \mathbb{I}_{\tilde{\lambda}^k = I^k(\lambda^k; \lambda^{k-1})},\\
&w_1((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i)\\
&= \begin{cases}
-\frac{q^{\lambda_j^{i-1} - \lambda_{j+1}^i + 1}(1 - q^{\lambda_j^i - \lambda_j^{i-1}})}{(1 - q^{\lambda_j^{i-1} - \lambda_{j+1}^i + 1})(1 - q^{\lambda_{j-1}^{i-1} - \lambda_j^{i-1}})}, \\
\qquad \text{if } \tilde{\lambda}^{i-1} = \lambda^{i-1} + \mathbf{e}_j, \tilde{\lambda}^i = \lambda^i + \mathbf{e}_{j+1} \text{ for some } j \ge 2;\\
-\frac{q^{\lambda_1^{i-1} - \lambda_2^i + 1}(1 - q^{\lambda_1^i - \lambda_1^{i-1}})}{1 - q^{\lambda_1^{i-1} - \lambda_2^i + 1}}, \\
\qquad \text{if } \tilde{\lambda}^{i-1} = \lambda^{i-1} + \mathbf{e}_1, \tilde{\lambda}^i = \lambda^i + \mathbf{e}_2;\\
1 + \frac{q^{\lambda_j^{i-1} - \lambda_{j+1}^i + 1}(1 - q^{\lambda_j^i - \lambda_j^{i-1}})}{(1 - q^{\lambda_j^{i-1} - \lambda_{j+1}^i + 1})(1 - q^{\lambda_{j-1}^{i-1} - \lambda_j^{i-1}})}, \\
\qquad \text{if } \tilde{\lambda}^{i-1} = \lambda^{i-1} + \mathbf{e}_j, \tilde{\lambda}^i = I^j(\lambda^i; \lambda^{i-1}) \text{ for some } j \ge 2;\\
1 + \frac{q^{\lambda_1^{i-1} - \lambda_2^i + 1}(1 - q^{\lambda_1^i - \lambda_1^{i-1}})}{1 - q^{\lambda_1^{i-1} - \lambda_2^i + 1}}, \\
\qquad \text{if } \tilde{\lambda}^{i-1} = \lambda^{i-1} + \mathbf{e}_1, \tilde{\lambda}^i = \lambda^i + \mathbf{e}_1;\\
0, \qquad \text{otherwise,}
\end{cases}\\
&w_2((\lambda^{i-1}, \lambda^i, \tilde{\lambda}^{i-1}), \tilde{\lambda}^i) = \mathbb{I}_{\tilde{\lambda}^i = \lambda^i}.
\end{aligned} \tag{5}$$

When $q \to 0$, this algorithm is reduced to the usual column insertion. However, note that $A_{1:n}$ is quite different from the q-column insertion discussed in the present paper. For example, when $q \in (0, 1)$, the weights can be negative and, therefore, it cannot be viewed naturally as a randomised algorithm in a probabilistic sense. This is also the case for the two algorithms called 'q-Whittaker-multivariate "dynamics" with deterministic long-range interactions' discussed at the end of this section. Moreover,

we have pointed out the dynamical duality between the q-column insertion and the q-row insertion, which is an analogue of the dynamical duality between the usual insertion algorithms. Finally, we can see from the weight function formulae in (5) that $A_{1:n}$ has row insertion like branching of inserting a displaced letter into a lower row, which never happens in the usual column insertion where a displaced letter stays in the same row or is inserted to a higher row. Therefore, we regard the q-column and q-row insertion algorithms as the most natural q-generalisations of the usual RS algorithms.

The natural question of whether the symmetry property holds for algorithms with general h has a negative answer, even when $q \to 0$. For example, let us take $n = 3$, $h = (1, 2, 1)$ and $\sigma = 231$. Inserting σ gives us

$$P = \begin{matrix} 1 & 2 \\ 3 \end{matrix}, Q = \begin{matrix} 1 & 2 \\ 3 \end{matrix}$$

while inserting $\sigma^{-1} = (3, 1, 2)$ gives

$$P = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}, Q = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}.$$

In terms of Proposition 6, these algorithms violate condition (iii) and (iv). It seems likely that an algorithm indexed by h has the symmetry property if and only if h is $(1, 1, \ldots, 1)$ or $(1, 2, \ldots, n)$.

In [2] another family of 'RSK-type', q-weighted algorithms associated with vectors h called 'q-Whittaker-multivariate "dynamics" with deterministic long-range interactions' were introduced. We omit the definitions here and point interested readers to Sect. 8.2.2 of that paper. When $h = (1, 2, \ldots, n)$ and $h = (1, 1, \ldots, 1)$, they also satisfy the conditions in Proposition 6 and thus enjoy the symmetry property.

Acknowledgments The author would like to thank Neil O'Connell for guidance. Research of the author was supported by EPSRC Grant No. EP/H023364/1.

References

1. de Beauregard Robinson, G.: On representations of the symmetric group. Am. J. Math. **60**(3), 745–760 (1938)
2. Borodin, A., Petrov L.: Nearest neighbor Markov dynamics on Macdonald processes. (2013) arXiv: 1305.5501
3. Burge, W.H.: Four correspondences between graphs and generalized Young tableaux. J. Combin. Theory **17**(1), 12–30 (1974)
4. Etingof, P.: Whittaker functions on quantum groups and q-deformed Toda operators. Transl. Am. Math. Soc. **2**(194), 9–26 (1999)
5. Fomin, S.: Two-dimensional growth in Dedekind lattices. PhD thesis, MS thesis, Leningrad State University (1979)

6. Fomin, S.: Generalized Robinson–Schnested–Knuth correspondence. Zapiski Nauchn. Sem. LOMI **155**, 156–175 (1986). In Russian
7. Fomin, S.: Duality of graded graphs. J. Algebr. Comb. **3**(4), 357–404 (1994)
8. Fomin, S.: Schensted algorithms for dual graded graphs. J. Algebr. Comb. **4**(1), 5–45 (1995)
9. Fulton, W.: Young Tableaux. London Mathematical Society Student Texts, vol. 35. Cambridge University, Cambridge (1997)
10. Gerasimov, A., Lebedev, D., Oblezin, S.: On a classical limit of q-deformed Whittaker functions. Lett. Math. Phys. **100**, 279–290 (2012)
11. Greene, C.: An extension of Schenstedś theorem. Adv. Math. **14**(4), 254–265 (1974)
12. Johansson, K.: Shape fluctuations and random matrices. Commun. Math. Phys. **209**(2), 437–476 (2000)
13. Knuth, D.E.: Permutations, matrices and generalized Young tableaux. Pac. J. Math. **34**, 709–727 (1970)
14. Kostant, B.: Quantization and representation theory. In: Proceedings of the SRC/LMS Research Symposium on Representation Theory of Lie Groups Oxford, England, pp. 287–316 (1977)
15. MacDonald, I.: Symmetric Functions and Hall Polynomials. Oxford Mathematical Monographs. Clarendon, New York (1998)
16. O'Connell, N.: Conditioned random walks and the RSK correspondence. J. Phys. A **36**(12), 3049–3066 (2003)
17. O'Connell, N.: A path-transformation for random walks and the Robinson-Schensted correspondence. T. Am. Math. Soc. **355**(9), 3669–3697 (2003)
18. O'Connell, N.: Directed polymers and the quantum Toda lattice. Ann. Prob. **40**(2), 437–458 (2012)
19. O'Connell, N., Pei, Y.: A q-weighted version of the Robinson–Schensted algorithm. Electron. J. Prob. **18**(95), 1–25 (2013)
20. Ruijsenaars, S.N.M.: Relativistic Toda systems. Commun. Math. Phys. **133**(2), 217–247 (1990)
21. Sagan, B.E.: The Symmetric Group. Graduate Texts in Mathematics, vol. 203. Springer, New York (2000)
22. Schensted, C.: Longest increasing and decreasing subsequences. Can. J. Math. **13**, 179–191 (1961)
23. Stanley, R.: Enumerative Combinatorics. Cambridge Studies in Advanced Mathematics, vol. 2. Cambridge University, Cambridge (2001)
24. Sasamoto, T., Wadati, M.: Exact results for one-dimensional totally asymmetric diffusion models. J. Phys. A. **31**, 6057–6071 (1998)
25. Viennot, G.: Une forme géométrique de la correspondance de Robinson-Schensted. In: Combinatoire et représentation du groupe symétrique, vol. 579, pp. 29–58. Springer, New York (1977)